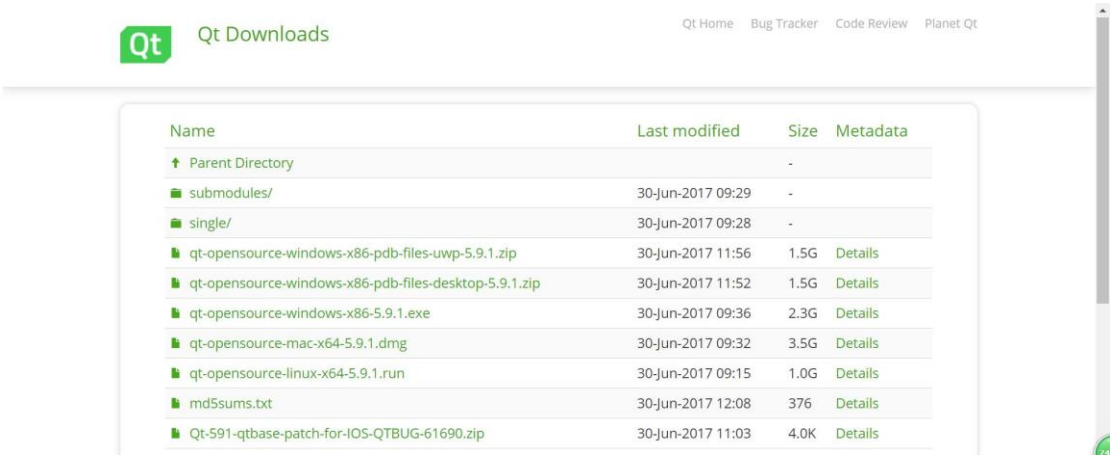
扑克牌游戏项目设计说明

一、项目介绍

- 本项目打算设计一个扑克牌游戏系统，能够实现两种扑克游戏（21 点和四人斗地主）的正常游玩，并且能够实现单机对战和局域网对战的功能。
- 我们主要的实现平台在 windows10 操作系统上。为加强用户体验，我们采用全图形界面设计，方便程序与用户交互。
- 该项目的图形化主要采用 Qt5.9.1, MSVC 2017 64bit 编译器实现。
- 本组统一使用 IDE：QT Creator 4.3.1(Community)
- 为了便于方便用户更加快速地安装 QT 并配置编译环境。本组将配置过程中应用到的文章及博客链接呈现如下：
  [Qt 5.9.1 + Visual Studio 2017 环境配置]
  (https://www.jianshu.com/p/a81350d630dd)

Qt 5.9.1 安装

①在 Qt 官网(http://download.qt.io/archive/qt/5.9/5.9.1/)下载 qt5.9.1



②选择“qt-opensource-windows-x86-5.9.1.exe”下载（windows 系统）。一路安装，一般不会有什么问题。注意选择编译器的时候选择 msvc2017。
③安装成功后，在工具-选项中-构建与运行-构建套件(Kit)中检查编译器前是否有红色感叹号标志。如有，则需手动配置 VS 编译器的位置，以本人的路径配置为例：

④有个要注意的地方是，qt creator 编译一遍生成新的一个文件夹（如“build-LandLord-Desktop_Qt_5_9_1_MSVC2017_64bit-Release”),如果有使用图片或声音文件之类的资源文件，要把资源文件放在生成的那个文件夹里面，不能放在.pro 在的文件夹里面，不然会无法找到。一般来说资源文件应当放在生成文件夹目录下，或者使用相对路径（如 image 文件夹下，"./images/LandLord.png")。

⑤如果要使用 Microsoft Visual Studio 2017 进行编译，需要先在工具-扩展和更新-联机中搜索“Qt Visual Studio Tools”并安装。重启程序使组件生效后，在菜单栏中的 Qt VS Tools-Qt Options-Qt Versions 窗口中点击 Add, Version Name 中填写 msvc2017_64(可以自定义)， 然后选择 Qt 路径(如本人的路径是“E:\Qt\Qt5.9.1\5.9.1\msvc2017_64”)。完成后，点击 Qt VS Tools-Launch Qt Designer，成功运行则说明配置完成。

二、需求分析

扑克游戏由于其小巧轻便，玩法多样的特性，成为人类史上最成功的娱乐工具之一。打扑克在中国亦是一种非常普遍的活动。随着信息技术的发展，扑克游戏借助网络游戏平台更是风靡全国。以斗地主为例，由于其独特的趣味性和益智性，我们可以发现各大棋牌游戏平台上都有“斗地主”游戏的踪迹。而 Windows 系统作为人们日常使用最多最广的系统，基于 Windows 系统的单机甚至联机游戏开发具有很大的实用价值和实际意义。

1. 扑克游戏的要求

1.1 友好的用户图形界面

友好的图形界面能够满足玩家的生理需求，这主要由游戏美术和音乐的策划

满足。除了布局、图片选择和音效要让用户舒适以外，图形界面必须保证玩家身份明确，点牌出牌功能便于操作和理解，玩家手牌或出牌记录明确。对用户的误操作必须及时地探知并处理，不能出现程序崩溃。

1.2 支持至少两种扑克牌游戏

扑克牌系统应能顺利兼容多种游戏，保证游戏的多样性。

1.3 可支持多位（四位）玩家游戏

扑克牌游戏的重要乐趣之一在于其中对抗性游戏的实现。对抗类扑克游戏给用户带来的核心体验是一种成就感，即胜利时刻的喜悦。多位玩家的参与能够迅速提升用户游戏体验。不管是出于运气或是技术而导致胜利或者失败，这给多位玩家的互动提供了可能，这也是扑克游戏最初作为一种互动游戏的价值所在。

1.4 可由 AI 扮演玩家角色

实际游戏操作时，往往会出现人类玩家不足的情况，因此一个执行效率高，能够模仿中等水平玩家思维方式的 AI 就比较重要。AI 需要整合一些常用的出牌套路（如拆牌算法），赋予 AI 一定的手牌评价标准。AI 的存在也能够使得用户能够“自娱自乐”，甚至能生成自动化测试方案方便程序员维护代码。

1.5 支持多人局域网游戏

局域网能够在小区域内将各种通信设备互连在一起，传输速率高，组网方便，使用灵活。在扑克游戏中整合局域网功能，能够实现小范围内人类玩家共同参与游戏，更方便实现不同人类玩家间的互动。

2. 扑克游戏的功能

2.1 21 点

21 点流程简单，只需点击“Hit”和“Stand”两个按钮即可进行游戏（牌堆用完后点击“Shuffle”更新牌堆，点击“Begin”初始化牌堆和积分）。和游戏流程有关的基本功能如下：

①在本游戏中，程序初始化时显示所有用户的头像和初始积分。并且整个游戏流程中玩家名字、积分和头像会一直显示。

②本游戏设计了单人游戏和多人游戏的功能，设计了简单的 AI。AI 自动完成 HIT,STAND 操作。在人类玩家的轮次则在其面前显示 HIT,STAND 按钮，等待人类玩家操作。所有玩家的手牌都会及时显示。

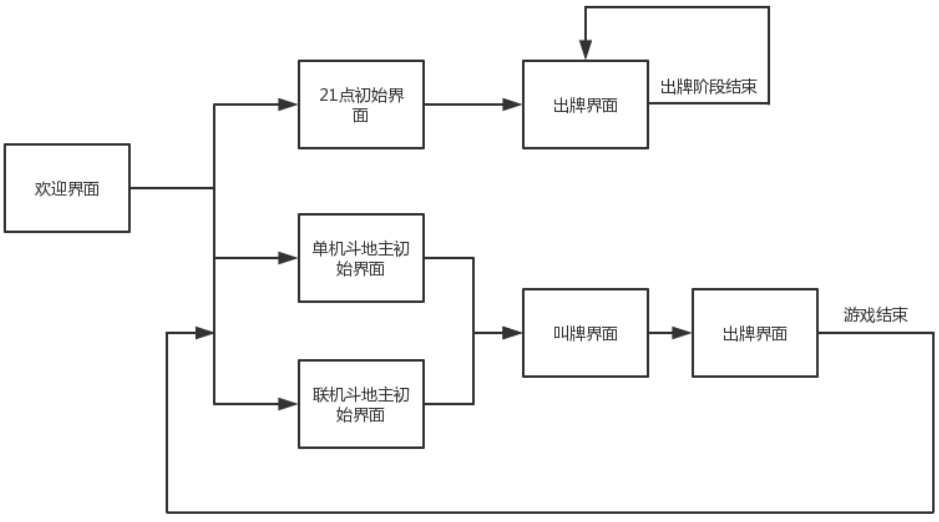
③规则判定。若牌面总和已超过 21 则提示“Bust”,每一轮游戏结束时显示每位玩家的胜负情况，并对金钱(积分)进行结算，胜+50，负-50，平则不变。

④牌堆耗尽时自动更新牌堆，也可以手动更新牌堆(Shuffle)。

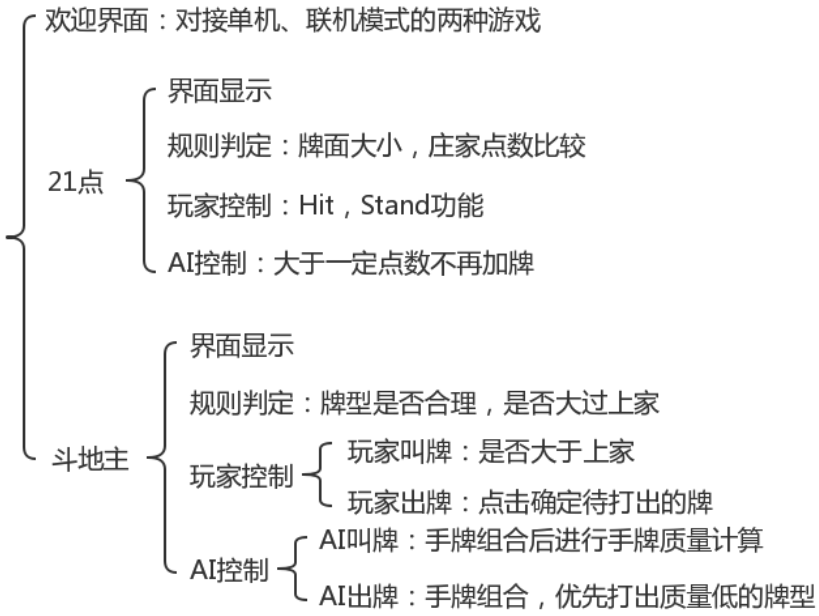
2.2 四人斗地主

除了基本的手牌点击选择功能，显示每位玩家上回合打出的牌及待压的牌以外，和游戏流程有关的基本功能如下：
①在本游戏中，程序初始化时显示所有用户的初始积分。此后每局游戏结束都会显示每位玩家的积分变化。
②在本游戏中，每局游戏结束都可设置每位玩家是否由电脑控制，并在游戏进程中实时显示每位玩家是由人类控制还是电脑控制。
③在本游戏中，设置了叫地主功能（1 分、2 分、3 分、不叫)。叫分必须高于之前玩家叫的最大分数或不叫，若有玩家叫 3 分则直接结束叫牌阶段，该玩家成为地主并获得底牌。
④本游戏设计了自动出牌及叫牌的 AI 算法。若玩家由电脑控制，会自动将手牌分为各种牌型，并优先打出面值较小的牌。会自动计算手牌质量以决定是否叫分。会自动打出大过上家的牌。
⑤玩家由人类控制时，设置了出牌和不出的基本功能。会自动对出牌是否合理进行判断，未大过上家或牌型不合理则阻止出牌。若玩家是第一个出牌(游戏开始或其他三位玩家均选择不出牌时)，使“不出”按钮失效。
⑥为了实现多人单机功能，每位玩家(包括 AI 玩家)出牌时，该玩家视角转为第一视角(AI 玩家手牌不显示)。若人类玩家数量≥2，则人类玩家的出牌阶段开始前，上一位玩家出牌结束后，会隐藏上一位玩家的手牌，弹出提示框“人类玩家即将出牌！”，确认后下一位玩家继续游戏，以防止手牌被看光。
⑦用明显的地主卡通头像及农民头像显示每位玩家的身份，使用了简单的出牌和不出音效，用背面朝上的卡牌数显示每位玩家的手牌数量。

2.3 游戏流程及功能图



三、总体设计



1. 欢迎界面

欢迎界面需要有设计精良的画面，能够顺利与各种游戏对接即可。在欢迎界面上能够分别选择 21 点单机版，21 点联机版，斗地主单机版，斗地主联机版即可。具有类似游戏大厅的作用。

2. 21 点游戏

2.1 界面显示

游戏进程中显示当前玩家的手牌，是否 Hit/Stand。只需在游戏结束时显示四名玩家拥有的手牌即可，及是否胜利即可。

2.2 规则判定

判定玩家的牌面大小（是否 BUST），并在选择完成后与庄家点数进行比对（以判定胜利或者失败）。

2.3 玩家控制功能

只需点击 Hit, Stand 两个按钮来控制是否加牌或不加。

2.4 AI 控制功能

大于一定点数则不再加牌，否则持续加牌。

3. 四人斗地主游戏

斗地主的游戏规则和流程相比于 21 点要复杂一些，因此需将游戏界面显示、游戏规则判定、AI 算法分开处理。

3.1 界面显示

包括玩家卡牌的实时显示以及点击功能，每位玩家上一次打出的牌的实时更新(或者 PASS)，叫牌、出牌、不出按钮功能，底牌的显示等。每局游戏结束时积分的显示，以及使得多人单机功能能够实现的界面。只显示当前玩家的手牌，其他玩家（或 AI 玩家）的手牌背面朝上，只显示数量。

3.2 规则判定

叫牌阶段，只需控制玩家叫分必须大过上家（或不叫），直至有玩家叫了 3 分或四名玩家均完成了叫牌。出牌阶段，先判断玩家/AI 出牌的牌型是否合理。若有上家，则判断牌型是否大过上家，是则成功出牌，否则令玩家重新出牌。当某一玩家当前手牌数为 0 时结束此轮游戏并进行结算。

3.3 玩家控制

3.3.1 玩家叫牌

不能叫不大于上家的分（除非不叫），只需令相关叫分按钮失效即可。

3.3.2 玩家出牌

点击手牌以确定打出的牌。出牌或不出按钮即为确认键，之后传入规则判定模块进行牌型计算。无上家时不出按钮失效。

3.4 AI 控制

3.4.1 AI 叫牌

先对自己的手牌进行组合，估计手牌大小以及至少需要几手出完，以此计算

手牌质量来决定要叫几分。对各种牌型赋予一定的权值，利用权值和出牌回合数量化手牌质量。

3.4.2 AI 出牌

在无上家的情况下，直接对手牌进行组合，然后打出质量（权值）最低的牌。在有上家的情况下，先对手牌进行初步组合，在大过上家的前提下打出质量最低的牌。若没有大过上家的牌（牌型不符），则进一步组合，再计算是否有大过上家的牌。没有则 PASS。

四、系统模块说明

1. 21 点

1.1 21 点主菜单界面——class welcome

class welcome 继承自 Qt 的 QWidget 类。主菜单实现了创建单机本地游戏、创建单人游戏、创建双人游戏、创建三人游戏、创建四人游戏、进入联机大厅、修改个人信息、查看游戏帮助以及退出游戏这些功能。以下是界面生成和交互的几个重要的成员和成员函数。

1.1.1 welcome::welcome(QWidget *parent):QWidget(parent),ui(new Ui::welcome)

welcome 的构造函数用于生成和初始化主菜单界面中的创建单机本地游戏、创建单人游戏、创建双人游戏、创建三人游戏、创建四人游戏、进入联机大厅、修改个人信息、查看游戏帮助、退出游戏的按钮以及返回上级的按钮（这里按钮指的是 QPushButton，下同）并将它们显示，与此同时生成和初始化修改个人信息和查看游戏帮助的页面（相关控件隐藏）。

1.1.2 void welcome::newthis4man()

newthis4man 用于将当前主菜单关闭（隐藏），与此同时生成一个单机本地四人游戏的窗口，即，实例化一个 form4 类（下文有所提及），与此同时将这个窗口显示出来。

1.1.3 void welcome::helpshow()

helpshow 用于将当前主菜单中所有按钮隐藏，显示帮助页面的图片，并显示出返回上级的按钮，通过点击这个按钮可以使得帮助页面的图片及按钮隐藏，并使得之前被隐藏的按钮再次显示出来，从而使用户再次使用主菜单。

1.1.4 void welcome::sethost()

sethost 用于将当前主菜单中所有按钮隐藏，显示出创建单人游戏、创建双人游戏、创建三人游戏、创建四人游戏，以及返回上级的按钮，通过点击创建游戏的按钮可以将主菜单隐藏并实例化相关的类（单人——rform0，双人——rform1，三人——rform2，四人——Form），通过点击返回上级按钮将这些按钮隐藏，同时显示之前被隐藏的按钮。

1.1.5 void welcome::solohost()

solohost 用于将当前页面隐藏并实例化一个 rform0 类（单人游戏，后续有所提及），并将之显示出来。

1.1.6 void welcome::brohost()

brohost 用于将当前页面隐藏并实例化一个 rform1 类（双人游戏，后续有所提及），并将之显示出来。

1.1.7 void welcome::triplehost()

triplehost 用于将当前页面隐藏并实例化一个 rform2 类（三人游戏，后续有所提及），并将之显示出来。

1.1.8 void welcome::grouphost()

grouphost 用于将当前页面隐藏并实例化一个 Form 类（四人游戏，后续有所提及），并将之显示出来。

1.1.9 void welcome::findgame()

findgame 用于将当前页面隐藏并实例化一个 host 类（联机大厅，后续有所提及），并将之显示出来。

1.1.10 void welcome::setself()

setself 用于隐藏当前主菜单中所有按钮，并显示用户的头像、昵称，以及修改头像和昵称的控件。这里将修改头像的控件设置为 6 个按钮，用户点击相应头像可将头像变更为相应的头像，与此同时将修改昵称的控件设置为一个 LineEdit（单行文本编辑框），点击 ok 按钮完成修改，若昵称不合法则会有控件提示错误信息，否则在用户昵称一栏会显示新的昵称。

1.2 21 点卡牌——class card 和 class hand

class card 是游戏中的扑克牌。class hand 指的是游戏过程中各角色（含牌堆）所持有的牌，因此它有子类 class player，class house，class deck。以下是它们的几个重要的成员和成员函数。

1.2.1 class card

class card 有成员 m_rank（大小）、m_suit（花色），以及一个指示是否正面向上的 bool 型变量，以及两个成员函数（详见下文）。

1.2.2 int card::GetValue() const

用于获取 card 类的牌面大小。

1.2.3 void card::Flip()

用于将 card 类进行翻面，即改变那个指示是否正面向上的 bool 型变量（用于庄家翻牌）。

1.2.4 int hand::GetTotal() const

获取某一角色（庄家、闲家，以及牌堆）的手牌总点数。

1.2.5 void hand::Clear()

用于清空某一角色的手牌。

1.2.6 void Deck::Populate()

用于生成一副牌堆。

1.2.7 void Deck::Shuffle()

用于打乱牌堆的顺序，即，洗牌。

1.2.8 void Deck::Deal(hand& ahand)

用于将牌堆的一张牌给予某个角色（庄家、闲家或牌堆），实现发牌、叫牌的功能。

1.2.9 void Player::Bust()

结算闲家爆牌事件，这里表现为扣闲家的钱。

1.2.10 void Player::Win()

结算闲家获胜事件，闲家失败或平手也与此类似。

1.2.11 bool House::IsHitting() const

实现庄家的根据手牌点数大小而决定是否叫牌的功能。

1.2.12 void House::FlipFirstCard()

将庄家的第一张手牌翻面，以符合游戏的规则。

1.3 21 点客户类——class client

class client 是游戏的客户类。用于用户设置自己的昵称和头像，因此显然这个类拥有记录用户昵称的成员 c_name(QString 型变量，默认为“Dallas”)和记录用户头像的成员 c_ava(int 型变量，默认为 1)，以下是该类的全部成员函数。

1.3.1 void client::setname(QString n)

用于改变用户的昵称。

1.3.2 void client::setava(int a)

用于改变用户的头像。

1.4 21 点本地四人游戏界面——class form4

class form4 继承自 Qt 的 QWidget 类。用于进行本地的四人游戏。以下是界面生成和交互的几个重要的成员和成员函数。

1.4.1 form4::form4(QWidget *parent):QWidget(parent),ui(new Ui::form4)

用于生成和初始化本界面中的开始游戏、继续下轮、退出游戏、叫牌、停牌等按钮，以及显示各闲家牌面、个人信息、钱数、手牌点数的控件。建立这些控件和一些成员函数的连接关系，从而确保用户的交互。

1.4.2 void form4::play()

用于确保游戏的进程，即进行发牌，并显示各玩家（含庄家）的手牌，显示第一顺位闲家叫牌或停牌的按钮，等待该闲家决定。

1.4.3 void form4::begin()

用于生成一副牌堆，并显示各闲家的昵称、钱数、头像，并开始一轮游戏（利用成员函数 play）。

1.4.4 void form4::shuffle()

用于利用当前牌堆进行下一轮游戏（与名字不同），是开始下一轮。当没有生成牌堆时，本函数视为成员函数 begin。

1.4.5 void form4::p1hit()

处理第一顺位闲家的叫牌事件，其他顺位的闲家类似，在此不再赘述。

1.4.6 void form4::p1stay()

处理第一顺位闲家的停牌事件，其他顺位的闲家类似，在此不再赘述。

1.4.7 void form4::endgame()

处理庄家的回合（在第四顺位的闲家停牌或爆牌后），庄家展示手牌后进入到本轮游戏的结算页面，公布所有闲家的胜负结果。

1.4.8 void form4::showPlayer1cards(Player &aGenericPlayer)

用于展示第一顺位闲家的手牌，其他顺位闲家类似，在此不再赘述。

1.5 21 点联机大厅——class host

class host 继承自 Qt 的 QWidget 类。实现了一个类似于 CS1.6 或红警那样的联机大厅界面，用户可以通过此联机大厅获取局域网内主机的数量、姓名、主机用户的头像、主机内当前人数及人数上限。以下是界面生成和交互的几个重要的成员和成员函数。

1.5.1 host::host(QWidget *parent):QWidget(parent),ui(new Ui::host)

用于生成和初始化联机大厅界面，并实例化一个 QUdpSocket 类，将退出联机大厅、刷新、连接主机的按钮与下述成员函数相结合。

1.5.2 void host::on_freshbtn_clicked()

用于刷新联机大厅列表（先清空），再通过 UDP 协议在全局域网内广播字符串“f”。

1.5.3 void host::read_data()

用于读取并处理在局域网内接收到的字符串（主机接受字符串“f”或“c”后，会发出主机的信息），进而显示主机的状况（主机用户信息，以及主机的当前人数和人数上限），将之添加到联机大厅列表当中。

1.5.4 void host::on_listWidget_itemClicked(QListWidgetItem *item)

通过点击联机大厅列表中的某大厅，获取其主机的 IP 地址，并将此地址储存以方便后续联机。

1.5.5 void host::on_pushButton_clicked ()

在已选中某主机的情况下（未选中则无用），向该主机发送“c”字符串，表示联机请求。进而通过成员函数 cconnected 进行联机。

1.5.6 void host::cconnected()

根据主机的回应接收用户处于的顺位，根据不同的顺位实例化不同的类（第一顺位——Form1，第二顺位——Form2，第三顺位——Form3），若顺位不当（人数过多）或主机掉线则提示相应错误信息。

1.6 21 点主机游戏——class Form

class Form 继承自 Qt 的 QWidget 类。用于进行联机的四人主机端游戏，类似的单人主机游戏（class rform0），双人主机游戏（class rform1），三人主机游戏（class rform2）在此不再赘述，与本地四人游戏（class form4）类似的功能也不再赘述，只介绍特色的功能。该类使用 TCP 协议和 UDP 协议进行联机，其中主要的成员和成员函数如下。

1.6.1 void Form::read_data()

用于阅读并处理通过 UDP 协议接受到的其他用户的刷新（“f”字符串）或联机请求（“c”字符串），给发信的 IP 地址发送该主机的用户信息以及当前人数、人数上限。

1.6.2 void Form::slot_newconnect()

用于创建新的连接（通过 TCP 协议），并更新该主机内玩家的数量。

1.6.3 void Form::begin()

由于联机游戏区别于单机游戏，主机与其他玩家的牌堆必须是统一的（不然乱了套了），这里采取的方式是主机洗牌后将自己的牌堆顺序发给其他用户，各用户采取本地结算的方式运行游戏。本成员函数相比本地四人游戏多了这一点功能。

1.6.4 void Form::slot_recvmessage()

用于处理其他用户的决策，这里规定为用户之间通过发送字符串而交流，某用户与主机交流，主机再将该用户的决策以特殊字母开头的字符串的形式发给其他用户，从而其他用户也能进行响应。

1.6.5 void Form::slot_disconnect()

用于处理其他用户掉线的情况，只要有一个用户掉线，本局游戏立刻终止，主机返回主菜单，其他用户返回联机大厅。

1.6.6 void Form::player4hit()

用于处理主机叫牌的情况，给其他用户发送特殊字母开头的字符串。停牌也是类似的，在此不再赘述。值得一提的是，本游戏支持少于四人的联机对战，由于 21 点游戏较为简单，这里设置为由计算机扮演的闲家在手牌点数小于 17 时不会停止叫牌（除牌堆为空时）。进行少于四人的游戏时，主机无需接受计算机扮演的闲家的指令，直接给其他闲家发送该顺位闲家的决策结果。

1.7 21 点客户端游戏——class Form1

class Form1 继承自 Qt 的 QWidget 类。用于实现处于第一顺位的闲家的联机对战，第二顺位（class Form2）和第三顺位（class Form3）类似，在此不再赘

述。以下是界面生成和交互的几个重要的成员和成员函数。

1.7.1 void Form1::slot_connected()

用于建立与主机的连接。

1.7.2 void Form1::slot_recvmessage()

用于处理主机发送的消息，主机可能发送的信息有：开始游戏（伴随着一副牌的顺序）、继续下轮、其他用户的决策（特殊字符开头）、其他用户的信息、主机的决策、主机用户的信息。需要对这些不同的信息采取不同的处理方法。

1.7.3 void Form1::slot_disconnect()

用于处理主机掉线的情况，这里设置为主机一旦掉线就返回联机大厅。值得注意的是，主机掉线也可能是被动的，是由另一其他用户掉线引起的连锁反应。

1.7.4 void Form1::player1hit()

用于处理闲家叫牌的情况，给主机发送特定的字符串。停牌的情况也类似，在此不再赘述

2. 四人斗地主

2.1 四人斗地主的界面及流程控制——Class MainWindow

Class MainWindow 继承自 Qt 的 QMainWindow 类。QMainWindow 作为 Qt 展示主界面的模板类，提供了一个可交互的界面和内部相应的功能，同时自带菜单栏等主界面常用的组件。四人斗地主的交互界面就建立在 QMainWindow 之上。

通过继承自 QMainWindow 的类 MainWindow，程序实现了对牌局的可视化和交互过程，以下是界面生成和交互的几个重要的成员和成员函数。

2.1.1 explicit MainWindow(QWidget *parent = 0)

MainWindow 的构造函数用于生成和初始化一些之后需要用到的全局变量以及成员，比如生成用于比对牌型的矩阵；同时生成整个游戏界面的背景，选择开始游戏的按钮，以及选择各个位置上是 AI 还是玩家的选项。

2.1.2 void on_ButtonConfirm_clicked()

玩家点击开始游戏按钮之后，程序进入按钮点击事件函数。

在该函数内部，首先程序依次进入 GameStart()、ShowLandLordRoundCurrentPlayer()函数，在这过程中程序将进行牌局的初始化，即洗牌、发牌，将当前玩家的初始牌型显示到界面上，并显示叫牌按钮。如果当前玩家是 AI，则程序将进入 AILandLordDecided()函数自动进行叫牌的流程，如

果是玩家，程序则等待玩家选择叫牌或 Pass，当玩家点击按钮后，程序进入 HumanLandLordDecided(int choice)函数。

2.1.3 void EndLandLordDecide()

当叫牌阶段结束，即所有玩家都进行过叫牌选择之后，程序进入该成员函数。在该函数中，程序会确定当局地主，当局的分数，将底牌发给地主，初始化之后打牌过程要使用到的成员和全局变量，将叫牌阶段的界面清除，当前玩家变更为地主，并进入 InitPlay()函数。

2.1.4 void InitPlay()

该成员函数将在叫牌彻底结束以及一名玩家出完牌之后执行，用来初始化下一位玩家的出牌阶段。函数将根据当前玩家重新生成当前界面，如果是 AI 将展示手牌背面并直接进入出牌程序，如果是人类玩家则进行提示、展示手牌并等待玩家选择要出的牌或 Pass。

2.1.5 void MainWindow::IndexCalculate()

该函数使用于玩家出牌阶段，用来根据玩家对牌的点击，实现牌被选中之后的位移以及将被选中的牌储存下来，用于接下来合法性的判断。

2.1.6 void MainWindow::AIPlayCard()、void MainWindow::HumanPlayCard()

该两个函数用来控制程序的出牌阶段，前者进行 AI 出牌的运算和判断，而后者则直接进入牌型合法性判断。

2.1.7 void MainWindow::JudegePlayedCard()

该函数是出牌合法性判断函数，用来判断所出的牌是否合法，并将判断的结果传递到下面提到的 RoundEnd(int problem)函数中，用来结束这一轮的出牌，如果玩家的出牌不合法，则会被要求重新出牌。

2.1.8 void RoundEnd(int problem)

人类玩家或 AI 出牌选择以及牌型判断结束之后将进入该函数，如果出牌错误，牌型违规，如果当前玩家是玩家则提示错误，如果是 AI 则直接跳过该轮。如果出牌正确，则清除上家出的牌，更新数据并重新进入 InitPlay()。

2.1.9 void GameOver()

当当前出牌玩家牌的数量变为 0，即意味着游戏结束，此时游戏首先弹框说明游戏胜利方并进入 GameOver()函数。该函数将进行分数的结算，初始化所有的参数并清除当前的界面并回到分配玩家的初始页面。

2.2 四人斗地主的牌堆管理——Class Pile、PlayerCard、Card、Choose

2.2.1 牌堆控制——Class Pile

程序使用类 Pile 来控制牌堆。牌堆的作用主要是在开局阶段，Pile 需要完成牌堆的初始化，洗牌以及将牌分配给各个玩家和底牌，这通过成员函数 Shuffle()以及 Allocate()来进行。类 Pile 中包含成员 Player 以及 HoleCard，用来存储分配的结果。

2.2.2 手牌、底牌控制——Class PlayerCard

程序使用类 PlayerCard 来储存、控制手牌和底牌，内置成员函数来进行手牌的排序和 AI 自动出牌，并记录该玩家的信息，比如是否是 AI，是否是地主并提供该手牌具有的牌型信息。

2.2.3 扑克牌信息储存——Class Card

类 Card 用来记录一张牌的基础信息——点数、花色和实际大小，并通过重载大于小于符号实现牌的排序。

2.2.4 上家出牌记录——Class Choose

类 Choose 继承自 PlayerCard，因此具有记录多张牌和洗牌的功能，同时，Choose 记录着当前出最大牌的玩家出的牌型和玩家序号，以此来判断是否过了一轮重新回到出牌玩家出牌，并结合类 MainWindow 内的成员函数来判断出牌是否合法。

2.3 四人斗地主的 AI 算法——class DividedCard

为了方便改进和维护 AI 算法，AI 程序是独立于牌堆和界面程序之外的。当需要使用 AI 时，将所有数据传入 class DividedCard 中单独计算，然后输出计算结果。每个 DividedCard 对应的是类 PlayerCard(但采用不一样的存储格式)。新定义了 struct CardGroupData 存储已经分类的牌型。以下是 AI 计算的几个主要函数或结构：

2.3.1 已经分类的牌型 struct CardGroupData

category 对应牌型编号，index 对应原手牌下标，MAXCard 对应决定牌型的大小的牌的点数（如三带二中则是三条的点数，连对中则是最大对子的点数），nValue 是按照一定规则计算出的牌的价值。该结构体重载了">"和"+=", 其中>构成了一个偏序关系，包括炸弹大于其他牌型及同类型牌之间点数比较。+=则用于牌型的组合，如三条和对子组成三带二，三个及以上对子组成连对等。同时，这个结构体也起着存储上家牌型的作用，作为比较。

2.3.2 void DividedCard::Interface(int pcase)

这个函数完成与外部函数对接的工作，pcase 用 0,1,2 分别代表叫牌计算，无上家出牌策略与有上家出牌策略。然后调用函数输入计算所需要的数据。

2.3.3 void DividedCard::CombinationStep()

在数据刚传入该结构体时，所有牌仍然以单牌的形式呈现。实际上DividedCard中共有4个CombinationStep()函数,分别对手牌进行四种组合(组合时调用Compose函数)。这四种组合分别是：①判断手牌中是否有王炸，有则组合;②找出手牌中所有能组成炸弹、三条、对子的组合;③组合出所有的三顺、连对、单顺;④找出所有的三带二组合（包括三顺带多个对子的组合)，这一步额外调用函数TripleWithDual。

2.3.4 void DividedCard::WeightConfirm()

这一函数与ValueJudge()函数是一体的，按照一定规则(炸弹权重大于其他牌型，其余牌型的权值由MAXCard决定)算出所有牌型的权值。接下来对接函数DetermineChoice()选出权值最小的牌打出(或者大于上家的权值最小的牌)，或者对接函数LandLordJudge()根据手牌总价值返回是否叫地主。

3. 欢迎界面

3.1 界面设计——主界面和设置界面

这部分的代码设计任务简单，只需要运用一些简单的信号和槽函数就可以实现。

3.1.1 主界面MenuPanel

MainPanel的构造函数用于生成相关的可视界面，界面需要用到的按钮主要调用gamelab类生成，布局采取垂直布局和水平布局交错的方式。而中部的跑马灯函数通过计时器QTime类每隔一段时间重新定义字符串的位置实现。这里采用的也是信号与槽的点击-响应模式。

3.1.2 设置界面settingpanel

玩家点击设置按钮之后，打开新的窗口，即设置界面。

设置界面的设计方法和主界面类似，调节音量采用Qt自带的QSlider类函数，对于音量调节的样式设计采取QSS设计（类似html的格式)，对于不同的状态进行不同的颜色样式设计。头像选择采用Qt自带的QFileDialog函数，该函数可以生成一个打开系统文件的对话框并返回文件路径。

3.2 按钮的实现——Class gamelabel

该函数主要用于生成对鼠标时间响应的按钮，给定鼠标移入和未移入两种状态，通过Qt自带的鼠标监视函数在不同的状态下，绘制不同的按钮。例如鼠标移入或者点击时绘制鼠标移入的按钮图片，鼠标移出时则绘制鼠标未移入时的按钮图片。

五、系统设计难点及其解决

1. 21 点游戏控制台与图形界面对接的问题

由于 21 点是本项目最先完成的游戏，在实现过程中有着突破性和先导性的意义，但在设计此子项目时也突现了在设计初期的困难——实现手牌和代码的对接。通过学习 Qt 基础知识，使用数量可变的 QLabel 存储牌面图片，解决了这一问题。

2. 局域网联机功能的实现

由于缺乏对于局域网通信的认识，在设计此功能时有着无从下手的困难。在此方面参考了一个局域网内使用 TCP 协议进行聊天的软件的代码，我们采取局域网内客户端和主机通过 TCP 协议发送特定字符开头的字符串的方式实现通信，二者再根据编码方式对字符串进行处理，进而实现联机的沟通。

3. 联机大厅功能的实现及断线处理

最初采取的联机方式是输入主机 IP 地址以及端口号，这样的方法不仅不美观，而且无法处理主机人数过多或其他客户端抢先联机的异常情况（无法确定玩家顺位）。这里通过 UDP 协议建立了一个联机大厅，确保了客户端连接主机的顺位正常，以及对于搜索到主机但主机稍后断线这种异常情况有所预防。增加了游戏时各玩家掉线时的处理，主机返回主菜单，各客户端返回联机大厅，防止异常情况的发生。

4. 斗地主游戏设计过程中的问题

由于斗地主游戏规则本身较为复杂，规则判定本身就成为一个难点。如何逐个实现叫牌功能、出牌功能，统一输入输出格式是编程中的重要问题。实际操作过程中出现过想打出的牌与实际出牌不符，当手牌快出完时出现未知错误的问题。最后统一采用数组下标实现出牌(用二维数组存储了 48 种牌型以供比对)，用相同点数的牌的数量对应牌型，并新设一个类（choose）存储上家牌型等实现了斗地主游戏各个功能之间的协调。

5. AI 算法的设计

如何设计一个能自动出牌的 AI 也是一个重要难点。21 点的 AI 比较简单，只需判断当前点数大小即可，斗地主 AI 则复杂得多。为了后期测试方便，我们把 AI 算法的设计放在比较靠前的阶段。实际测试中出现了不少 AI 出错牌（从而一直无法通过规则判定陷入死循环）的问题，包括牌型不对，出牌未大过上家的问题。这一步主要通过设置大量断点和中间输出找到 Bug 并进行逐步修正。

6. 游戏控制台与图形界面对接的问题

为了调试方便，后台程序员编写斗地主时使用控制台调试。在和图形前端对接的时候，由于部分游戏流程在 main 函数中实现，从而导致了一定的困难。最后程序员用继承的方式重新设计了 Qt 窗口类(未修改 AI 算法和与原牌堆有关的类的定义)，在牌堆类中添加了一些功能函数，并使用了原控制台输出函数辅助 Debug。

7. 斗地主点牌的问题

Qt 界面的 Label 虽然便于显示图片，但有时可能会有互相遮挡的问题。我们继承并重新设计了 QGroupBox 使之能响应鼠标点击事件。主要是添加了编号属性使得不同牌之间能够互相区分。

8. 用户界面设计的问题

21 点单机四人直接在四个座位上都设置“HIT”，“Stand”按钮即可。一般的斗地主一般都只有一个玩家的视角。为了使得多人单机功能实现，本游戏采用了旋转视角，并且人类玩家之间切换时会弹出提示框并隐藏当前手牌（防止不同玩家之间手牌透露），隐藏 AI 玩家的手牌。所有玩家轮流成为第一视角，并实时更新每位玩家打出的手牌信息，以及额外添加了“上家出牌”显示，让用户即使知道自己要压的是哪位玩家打出的什么牌。此外，Qt 界面的 Label 虽然便于显示图片，但有时可能会有互相遮挡的问题。我们继承并重新设计了 QGroupBox 使之能响应鼠标点击事件。主要是添加了编号属性使得不同牌之间能够互相区分。为了防止手牌遮挡，所有的手牌（QGroupBox）都是动态分配的。

9. 对接联网功能的问题

局域网对接期间可能要处理信号发送和接收的问题，如何传递洗牌、叫牌、出牌信号，处理断线问题是实现过程中的一个难点。我们最后归纳出了所有需要传递的信号种类，使用了 TCP 协议传格式化的字符串，用字符串存储洗牌信号(一个随机乱序的数组)，出牌信号(牌下标组成的数组)等。并由主机端(host)统一接收所有来自客户端的信号并发送给各个客户端。有客户端(或主机端)断线则会全员显示“Connection Down!”。

六、总结

本项目已成功实现 21 点及斗地主单机及局域网功能的实现，以及主界面和不同游戏的对接，按照说明书流程即可正确运行。刚开始做这个 project 的时候，本队队员遇到的最大问题就是感到无从下手，一头雾水。仅仅是看过书本和听老师讲解似乎不足以完成一个大程序。但是硬着头皮去做，边学边做，经过实操之后，再回过头看，发现原来不太懂的东西似乎就有些懂了。在不断探索并实现新功能的过程中，我们收获颇丰。

七、附录

1. 任务时间线

任务	Deadline	负责人
代码风格、游戏内容讨论	2019.5.1	全体成员
21 点游戏流程及 AI 编写	2019.6.1	后端程序员
斗地主游戏流程及 AI 编写	2019.6.1	后端程序员
图形化接口的确认与对接	2019.6.9	前端程序员
素材资源收集、UI 设计	2019.6.9	测试、设计岗位
联机功能的实现	2019.6.16	前端程序员
开发文档编写	2019.6.16	全体成员

2. 团队开发注意事项

①在图形化过程中，后端程序员需要和前端程序员有充分的交流，并确认函数的接口原型，避免产生接口冲突的情况。
②采用统一的代码风格，由全组人员讨论后制定，不得擅自更改，如果有分歧则

少数服从多数。
③代码要有充分的注释，选择有意义的变量名。
④两位后端程序员要先自行测试自己编写的功能函数，通过后相互交换测试，确认没有 Bug 后方可交给前端程序员。前端程序员在编写程序时可要求后端程序员陪同完成（解释函数功能或及时改掉函数中的 BUG,或根据实际情况改写函数）。
⑤保证按照时间线完成相应任务，不得拖延。

3. 人员职责

角色	姓名（学号）	职责
组长	杨嘉南（3160102138）	总策划，成员讨论组织，后端程序员、project 总体架构设计、斗地主游戏设计、报告撰写
组员	王麟瑞（3160101251）	后端程序员，IDE 选择、安装、测试，21 点游戏设计，网络编程实现
组员	沈哲宇（3160102114）	前端程序员，IDE 选择、安装、测试，图形和算法对接
组员	程明春（3170105695）	界面设计、素材收集、project 总体测试